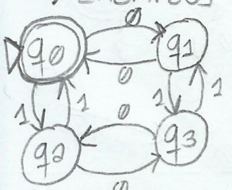
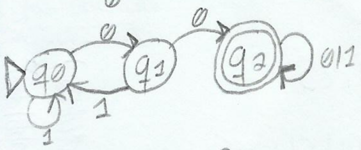
→ AUTÔMATOS FINITOS (AF)

É um modelo matemático para reconhecer linguagens regulares.
Podem ser: AFD, AFN, AFN-ε (AFN com ε-transições)
A transição de q_0 recebendo 0 e indo para q_1 é denotada por $\delta(q_0, 0) = \{q_1\}$

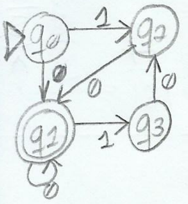
→ EXEMPLOS DE AF



Reconhece as sentenças com um número par de 0's e par de 1's.



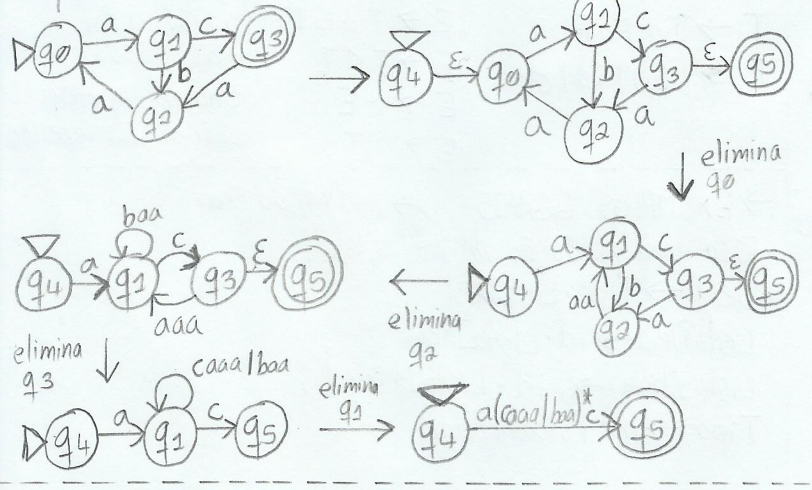
Reconhece as sentenças com dois 0's seguidos em alguma posição. Equivale a $(0|1)^*00(0|1)^*$.



Reconhece as sentenças cujo todas as substrings de 3 símbolos tem pelo menos dois 0s.

→ CONVERSÃO AF (AFD ou AFN) → ER | ELIMINAÇÃO DE ESTADOS

1. O estado inicial não pode ter transições chegando nele e nem ser final. Cria um novo estado inicial c/ transição vazia.
2. O estado final deve ser único e não pode haver transições saindo dele. Criado um novo estado final e puxadas transições ε de cada antigo estado final p/ o novo estado final.
3. Transições próprias transformam-se em fechos e transições paralelas em uniões.



→ OPERAÇÕES COM LINGUAGENS

- União: $L_1=\{a,ab\}$; $L_2=\{b,ab\}$; $L_1 \cup L_2=\{a,ab,b\}$
- Diferença: $L_1=\{a,ab\}$; $L_2=\{b,ab\}$; $L_1-L_2=\{a\}$
- Concatenação: $L_1=\{a,b\}$; $L_2=\{c\}$; $L_1L_2=\{ac,bc\}$
- Potenciação: $L=\{a,b\}$, $L^2=\{aa,ab,ba,bb\}$

→ NOTAÇÃO ALGÉBRICA DE LINGUAGENS

$\{10^n \mid n \geq 0\} = \{1, 10, 100, 1000, \ldots\}$

Se $V=\{0,1\}$, $\{x00y \mid x \text{ e } y \in V^*\} = \{00, 000, 0001, 11000, \ldots\}$

→ EXPRESSÕES REGULARES (ER)

É uma notação que descreve linguagens regulares a partir de símbolos do alfabeto e da sentença vazia combinados a partir do fecho, concatenação e união. Representadas por máquinas de estados finitos.

→ REGRAS ALGÉBRICAS PARA ER

Sejam L, M e N expressões regulares:

$L|M = M|L$; $(L|M)|N = L|(M|N)$; $(MN)|L \neq (M|L)(N|L)$

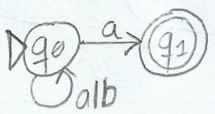
$(LM)N = L(MN)$; $LM \neq ML$; $L(M|N) = LM|LN$

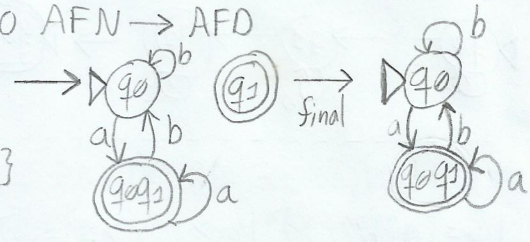
→ EXEMPLOS DE ER

$(0|1)^*0(0|1)(0|1)$ gera todas as sentenças em $\{0,1\}$ com no mínimo 3 símbolos e o antepenúltimo é 0.

$01(01)^*(0|\varepsilon) \mid 10(10)^*(1|\varepsilon)$ gera todas as sentenças em $\{0,1\}$ com 0s e 1s alternados.

→ CONVERSÃO AFN → AFD





$\delta(q_0,a)=\{q_0,q_1\}$

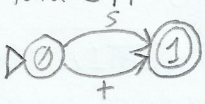
$\delta(q_0,b)=\{q_0\}$

$\delta(q_0q_1,a)=\delta(q_0,a) \cup \delta(q_1,a) = \{q_0,q_1\} \cup \{\varepsilon\} = \{q_0,q_1\}$

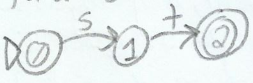
$\delta(q_0q_1,b)=\delta(q_0,b) \cup \delta(q_1,b) = \{q_0\} \cup \{\varepsilon\} = \{q_0\}$

→ CONVERSÃO ER → AFN-ε

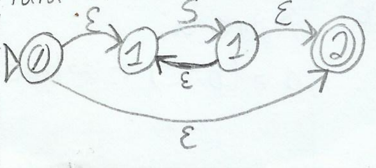
Para $s|t$



Para $s \cdot t$



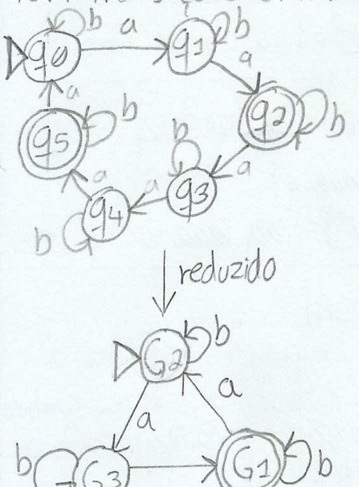
Para s^*



Slide 133
exemplo
colocar

→ Minimização de estados de um AFD

Divida o AFD em finais e não finais. Particione o grupo G em subgrupos de tal forma que dois estados S e T de G estão no mesmo subgrupo se e somente se para todos os símbolos de entrada A, os estados S e T têm transições em A para estados do mesmo grupo.



$\delta(q_2, a) = q_3 \in NF$
$\delta(q_2, b) = q_2 \in F$
$\delta(q_5, a) = q_0 \in NF$
$\delta(q_5, b) = q_5 \in F$ — G1

$\delta(q_0, a) = q_1 \in NF$
$\delta(q_0, b) = q_0 \in NF$ — G2

$\delta(q_1, a) = q_2 \in F$
$\delta(q_1, b) = q_1 \in NF$ — G3

$\delta(q_3, a) = q_4 \in NF$
$\delta(q_3, b) = q_3 \in NF$ — G2

$\delta(q_4, a) = q_5 \in F$
$\delta(q_4, b) = q_4 \in N$ — G3

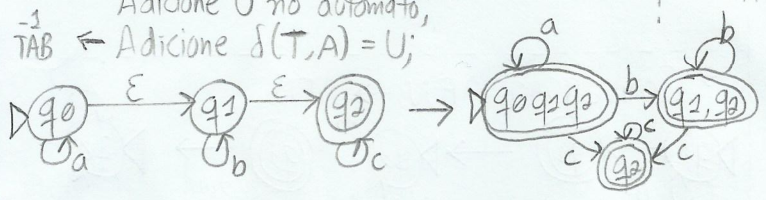
$F = \{q_2, q_5\}$; $NF = \{q_0, q_1, q_3, q_4\}$

→ Conversão AFN-ε → AFD

O primeiro estado do novo autômato é o fecho-ε do estado inicial. Fecho-ε é o conjunto dos estados alcançáveis através de transições-ε, incluindo o próprio estado atual.

Enquanto existe um estado T não marcado, faça:
- Marque T;
- Para cada símbolo de entrada A, faça:
  - U = fecho-ε(δ(T, A))
  - Se U não é estado do novo autômato, faça:
    - Adicione U no autômato;
  - TAB^{-1} ← Adicione δ(T, A) = U;

$\delta(q_0 q_1 q_2, a) = \{q_0\}$
fecho-ε$(\delta(q_0 q_1 q_2, a)) = \{q_0 q_1 q_2\}$



→ Construção de gramáticas livres de contexto

$L_9 = \{a^m b^m c^n d^n \mid m \geq 1, n \geq 1\} = ST$

S → aSb
S → ab
T → cTd
T → cd

$L_{10} = \{a^m b^n c^{n+1} d^{2m} \mid m \geq 1, n \geq 1\} = S$

S → aSdd
S → aXdd
X → bXc
X → bcc

$L_3 = \{a^i b^j c^k \mid j = i + k\} = TU$

T → aTb | ε ↘ $a^i b^i b^k c^k$
U → bUc | ε

→ Assossiatividade

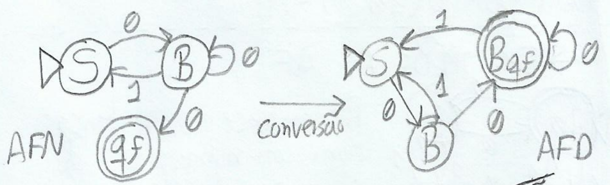
À esquerda se, ao se repetir, a operação se agrupa à esquerda:
(5-3) -2 ← 5-3-2

À direita:
a = (b=c) ← a=b=c

→ Conversão GR → AF

Cada não-terminal vira um estado; o estado inicial é o não terminal de início da gramática; crie um estado q_f que represente a aceitação. Para cada produção, faça:
A → aB gera δ(A, a) = B; A → a gera δ(A, a) = q_f; e
A → ε marca A como estado final também.

S → 0B
B → 0B
B → 1S
B → 0



(01|000*1)*000* ← conversão

→ Conversão GR → ER

Método da substituição

S → aA | ε
A → aS | bB
B → bS | cC
C → cS

B → bS | ccS
↓
A → aS | bbS | bccS
↓
S → $\overbrace{aa}^{\alpha_1}$S | $\overbrace{abb}^{\alpha_2}$S | $\overbrace{abcc}^{\alpha_3}$S | $\overbrace{\varepsilon}^{\beta_1}$ → (aa|abb|abcc)*
($\alpha_1|\alpha_2|..$)*($\beta_1|\beta_2..$)

→ Remoção da recursão à esquerda

A → Abc | Acd | cdB | C
B → Bc | dd

⇒

B → ddB'
B' → cB' | ε
A → cdBA' | cA'
A' → bcA' | cdA' | ε

→ Gramáticas ambíguas

Ocorre quando podemos construir mais de uma árvore de derivação p/ a mesma sentença. Remoção da ambiguidade:

E → E + T | T
T → T * F | F
F → (E) | id | cte

E → E + T | T	assos. à esquerda
E → T + E | T	asss. à dir.
E → T < T	não assos.
E → ~E	unários repetidos
E → ~T	unários não repetidos

→ Exemplos gerais

- Faça uma produção p/ int a,b,c; (→ ou float, ou char)

Decl → Tipo · ListaDecl
ListaDecl → id · ListaItem
ListaItem → , id · ListaItem | ;
Tipo → int | float | char